We Claim:

1. A method of processing fragments of a data packet comprising, for second and subsequent fragments of the data packet, rotating an EOF (end of fragment) portion of a payload of each fragment to before a SOF (start of fragment) portion of the payload of the fragment, the size of the EOF portion being equal to a size of a terminating portion of a respective preceding fragment.

2. A method according to claim 1 wherein the size of the EOF portion is a function of a sequence number of the fragment, the fragment size and the size of a smallest addressable memory unit.

3. A method according to claim 1 wherein the size of the EOF portion rotated expressed in symbols is given by

$$\text{Symbol\_rotation} = (SN*CS)\text{mod}(MUS)$$

where SN is a sequence number for the fragment, CS is a size of a fixed-size cell used to carry the fragment, in terms of symbols, and MUS is a size of a single memory location in a memory to which the fragments are to be transferred, also in terms of symbols.

4. A method according to claim 1 further comprising:

transmitting the fragments after rotation.

5. A method according to claim 1 further comprising:

receiving the fragments prior to rotation.

6. A method according to claim 1 further comprising:

transferring the first fragment and each other fragment thus rotated in sequence to an input buffer;

after any fragment is transferred to the input buffer, transferring the fragment to a packet buffer with no unaligned memory accesses for the first fragment, and a maximum of one unaligned memory access for each other fragment.

7. A method according to claim 6 wherein after a fragment is transferred to the input buffer, the fragment is stored in the input buffer in a first memory location, a plurality of intermediate memory locations and a last memory location, and wherein transferring each fragment to the packet buffer comprises:

for the first fragment, transferring the entire fragment to the packet buffer including a last portion of the fragment in a last written-to memory location in the packet buffer, the last portion being the terminating portion for the first fragment;

for second and subsequent fragments:

a) in an unaligned memory access, combining the portion of the preceding fragment in the last written-to memory location for the preceding fragment in the packet buffer with data from the first memory location and writing it to the last written-to memory location for the preceding fragment;

b) writing intermediate memory locations from the input buffer to the packet buffer using aligned memory accesses;

c) combining contents of the last memory location in the input buffer with the EOF portion for the fragment and writing to a last written-to memory location in the packet buffer for the fragment, the combination of the last memory location with the EOF portion for the fragment being the terminating portion for the fragment.

8. An apparatus for use in reassembling fragments of a data packet in memory having a smallest addressable memory unit, the apparatus comprising means for rotating an EOF (end of fragment) portion of a payload of each fragment to before an SOF (start of fragment) portion of the payload of the fragment and means for determining the size of each EOF portion as a function of the sequence number of the fragment, the fragment size and the size of the smallest addressable unit.

9. An apparatus comprising:

a packet rotator adapted to process fragments of a data packet comprising, for second and subsequent fragments of the data packet, by rotating an EOF (end of fragment) portion of a payload of each fragment to before a SOF (start of fragment) portion of the payload of the fragment, the size of the EOF portion being equal to a size of a terminating portion of a respective preceding fragment.

10. An apparatus according to claim 9 further comprising:

an input buffer;

a packet buffer having a smallest addressable memory unit;

the fragment rotator being adapted to transfer the first fragment and each other fragment thus rotated in sequence to the input buffer;

a buffer loader adapted to transfer contents of the input buffer to the packet buffer, by after any fragment is transferred to the input buffer, transferring the fragment to the packet buffer with no unaligned memory accesses for the first fragment, and a maximum of one unaligned memory access for each other fragment.

11. An apparatus according to claim 10 wherein the size of the EOF portion is a function of a sequence number of the fragment, the fragment size and the size of the smallest addressable memory unit.

12. An apparatus according to claim 11 wherein the size of the EOF portion rotated expressed in symbols is given by

$$\text{Symbol\_rotation} = (SN*CS)\bmod(MUS)$$

where SN is a sequence number for the fragment, CS is a size of a fixed-size cell used to carry the fragment, in terms of symbols, and MUS is the size of the smallest addressable memory unit.

13. An apparatus according to claim 10 adapted to transmit the fragments after rotation.

14. An apparatus according to claim 10 further adapted to receive the fragments prior to rotation.

15. An apparatus according to claim 10 wherein after a fragment is transferred to the input buffer, the fragment is stored in the input buffer in a first memory location, a

plurality of intermediate memory locations and a last memory location, and the buffer loader transfers each fragment to the packet buffer by:

for the first fragment, transferring the entire fragment to the packet buffer including a last portion of the fragment in a last written-to memory location in the packet buffer, the last portion being the terminating portion for the first fragment;

for second and subsequent fragments:

a) in an unaligned memory access, combining the portion of the preceding fragment in the last written-to memory location for the preceding fragment in the packet buffer with data from the first memory location and writing it to the last written-to memory location for the preceding fragment;

b) writing intermediate memory locations from the input buffer to the packet buffer using aligned memory accesses;

c) combining contents of the last memory location in the input buffer with the EOF portion for the fragment and writing to a last written-to memory location in the packet buffer for the fragment, the combination of the last memory location with the EOF portion for the fragment being the terminating portion for the fragment.

16. A method of processing fragments of a data packet comprising, for second and subsequent fragments of the data packet, shifting each fragment by an amount equal to a size of a terminating portion of a respective preceding fragment, the size of the shift being a function of a sequence number

of the fragment, the fragment size and the size of a smallest addressable memory unit;

transferring the first fragment and each other fragment thus shifted in sequence to an input buffer;

after any fragment is transferred to the input buffer, transferring the fragment to a packet buffer with no unaligned memory accesses for the first fragment, and a maximum of one unaligned memory access for each other fragment.

17. A method according to claim 16 wherein after a fragment is transferred to the input buffer, the fragment is stored in the input buffer in a first memory location, a plurality of intermediate memory locations and a last memory location, and wherein transferring each fragment to the packet buffer comprises:

for the first fragment, transferring the entire fragment to the packet buffer including a last portion of the fragment in a last written-to memory location in the packet buffer, the last portion being the terminating portion for the first fragment;

for second and subsequent fragments:

a) in a unaligned memory access, combining the portion of the preceding fragment in the last written-to memory location for the preceding fragment in the packet buffer with data from the first memory location and writing it to the last written-to memory location for the preceding fragment;

b) writing intermediate memory locations from the input buffer to the packet buffer using aligned memory accesses;

c) writing contents of the last memory location in the input buffer to a last written-to memory location in the packet buffer for the fragment